DOCUMENT RESUME

ED 294 735 SE 049 129

AUTHOR Moore, J. L.; Sleeman, D.
TITLE Enhancing PIXIE's Tutoring Capabilities.
INSTITUTION Aberdeen Univ. (Scotland). Dept. of Computing Science.
REPORT NO AUCS/TR8709
PUB DATE 88
CONTRACT MDA-903-84-K-0279
NOTE 40p.; For related documents, see SE 049 127-130.
PUB TYPE Reports - Descriptive (141) -- Reports - Research/Technical (143)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Algebra; *Computer Assisted Instruction; Computer Science; Computer Uses in Education; Foreign Countries; Mathematics Education; *Mathematics Instruction; *Mathematics Skills; Models; Secondary Education; *Secondary School Mathematics; *Tutoring
IDENTIFIERS *Mathematics Education Research

ABSTRACT

This paper discusses the overall design of the PIXIE Intelligent Tutoring System, and more specifically a series of recent enhancements. The original system has been implemented to involve three separate phases: the offline, or model generation phase; the online, or tutoring, phase; and the analysis phase. The offline phase, which is completed prior to any interaction with a student, involves the construction of a set of student models for a given domain. The online phase involves the tutorial interaction with a student, consisting of both diagnosis and remediation of errors. During the post-interaction analysis phase, undiagnosed errors are examined and, if consistent, added to the existing domain knowledge base. Four recent enhancements to the system are discussed, each arising from a shortcoming that was noted in the system as a result of student trials. Two of these additions to the system involve the diagnosis of errors, and two involve the remediation of errors. (Author/TW)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

Enhancing PIXIE's Tutoring Capabilities

J. L. MOORE

D. SLEEMAN

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

X This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

L. P. Ward

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

AUCS/TR8709

Enhancing PIXIE's Tutoring Capabilities

J. L. MOORE

D. SLEEMAN

Computing Science Department

University of Aberdeen

King's College,

Aberdeen, Scotland AB9 2UB

Acknowledgements: This work was carried out through the support of ARI/ONR contract number MDA 903-84-k-0297.

Abstract

This paper discusses the overall design of the PIXIE Intelligent Tutoring System, and more specifically a series of recent enhancements. The original system has been implemented to involve three separate phases: the offline, or model generation, phase; the online, or tutoring, phase; and the analysis phase. The offline phase, which is completed prior to any interaction with a student, involves the construction of a set of student models for a given domain. Considerable effort has been expended to ensure that these sets are complete and non-redundant. The online phase involves the tutorial interaction with a student, consisting of both diagnosis and remediation of errors. During the post-interaction analysis phase, undiagnosed errors are examined and, if consistent, added to the existing domain knowledge base. Four recent enhancements to the system are then discussed, each arising from a shortcoming that was noted in the system as a result of student trials. Two of these additions to the system involve the diagnosis of errors, and two involve the remediation of errors.

1. Introduction

PIXIE is an Intelligent Tutoring System shell that attempts to diagnose and remediate student errors in a particular domain (Sleeman, 1987). This system has been implemented as three separate subsystems. The first, the offline phase, generates models that incorporate typical bugs, or errors, in the domain to be tutored. These bugs have been collected through paper-and-pencil tests and interviews with students. The second, the online phase, uses the models previously generated in the offline phase to diagnose and remediate a particular student's errors. The overall structure of the PIXIE system is shown in Figure 1. The separation of the system into three phases allows for a quick response time during the actual tutoring session; however, it also means that PIXIE is only able to detect previously encountered bugs, or mal-rules. Implementing a system capable of diagnosing bugs that have not been encountered before is a research topic currently being pursued (Sleeman, 1982; Hirsh & Sleeman, 1985). Presently, unanticipated answers can be processed during the third phase, post-interaction analysis, and, if consistent, added to the domain knowledge base.

1.1 Offline Phase

The PIXIE tutoring system has been designed so that it can be used with numerous subject areas. Each domain is represented by a rulebase, or knowledge base, which contains, among other things, the correct rules and incorrect rules (i.e. malrules) for solving tasks. Sleeman (1987) discusses in detail the creation and use of knowledge bases for the PIXIE system.

The offline system attempts to generate a complete and non-redundant set of models, and to do this efficiently. One of the

problems in generating student models is the vast size of the search space (Sleeman, 1983; Sleeman & Smith, 1981). The offline system generates all correct models, all incorrect models by substituting malrules into the correct models, and further produces all significant orderings of rules and malrules. A model is an ordered sequence of rules and malrules. The algorithm for generating a complete and non-redundant set of models has three stages:

1. Start with the correct model for a given task type, substitute malrule variants in the model, and generate all combinations.
2. Create models to represent interactions between rules.
3. Eliminate rules that have been subsumed.

These three steps are briefly discussed below, for a more detailed description see Sleeman (1983).

GENERATE MALRULE MODELS

The algorithm begins with a template, which is the set of rules that correctly solve a particular type of task. For the domain of algebra, the first example in Figure 2 illustrates a sequence of rules that will correctly solve tasks of the form ax + b = c. All malrule variants must then be substituted in place of the correct rules. Figure 2 also shows the substitution of a malrule for a correct rule and the incorrect trace that results. In general, given the set of rules (r1, r2, r3) and malrules (mr2, mr3), in which mr2 and mr3 are incorrect versions of r2 and r3 respectively, the set of models created would be (r1 r2 r3), (r1 mr2 r3), (r1 r2 mr3), and (r1 mr2 mr3). [The distinction needs to be drawn between a model and trace. A trace is the sequence of rules that have been used to solve a task. Figure 2 illustrates several traces of algebra

problems. A model may contain rules not used in the solution of a particular task. For example, the model for the first example in Figure 2 may be expressed as (N-TO-RHS ADD SUBTRACT SOLVE), whereas the trace would be (N-TO-RHS ADD SOLVE), because the rule for subtraction, SUBTRACT, does not fire with this example.]

INTERACTION

The second step in the algorithm involves a consideration of an interaction among rules. Notice in the above list of models for the rules r1, r2 and r3 that the order of the rules does not vary, however, order is sometimes significant as the second example in Figure 2 illustrates. Only correct rules are used in this example, but because there is an interaction between the rules MULT and ADD an incorrect model results if their order is reversed. This is a convenient way of representing precedence requirements, which are important in many domains. As a more general example, if r1 and r2 interact, then the model (r1 r2 r3) will produce a different answer than (r2 r1 r3), given an appropriate task, and both models must be included in the final set of models. In addition, the malrule version of r2 (i.e., mr2) must also be taken into account. This would produce the set of models (r1 r2 r3), (r1 mr2 r3), (r1 r2 mr3), and (r1 mr2 mr3), (r2 r1 r3), (mr2 m1 r3), (r2 r1 mr3) and (mr2 m1 mr3).

SYNTACTIC SUBSUMPTION

The final step in the algorithm deals with the condition of subsumption between rules. Rule r1 is said to subsume r2, if the conditions under which r1 fires are a subset of r2's conditions. The effect of r1 being placed before r2 in a model is that r2 could never be activated (and so would never appear in the task trace). For example, suppose the model contains the rule called ADD, which adds two numbers, and

REARRANGE, which changes a task of the form a + b*x to b*x + a. In a model to solve the task 3 + 4*x = 8, if ADD appears before REARRANGE, REARRANGE will never be used in the solution of this task. ADD transforms 3 + 4*x = 8 to 7*x = 8, thus the condition for REARRANGE to fire is never met.

The idea of subsumption is used to eliminate rules that will never fire in certain models. Thus, subsumption reduces the number of rules in models, but does not eliminate complete models. Notice that subsumption can be determined by inspection of the conditions under which each rule operates, therefore this will be called syntactic subsumption. (This distinction is needed because another form of subsumption will be discussed later in the paper.)

In summary, the domain rulebase contains the templates to correctly solve each type of task that is to be tutored. The offline model generator uses this template in order to produce the complete set of models in which all meaningful orderings of rules are proposed. Additionally, it substitutes the appropriate malrules for their corresponding correct versions in each of the models. The resulting models are then applied to the set of tasks to produce correct and incorrect answers.

1.2 Online Phase

The output from the offline phase - the set of models and the answers produced by those models - is used in the online, or tutorial, phase of the system. It is during this stage of the system that both the diagnosis and remediation of errors occurs. The remedial system provides two basic types of remediation: model-based remediation (MBR) and reteaching. Model-based remediation comments on the specific error(s) made by the student before presenting the correct method for solving the task. The

diagnosis of a student's error(s) is necessary for MBR. An example of MBR is provided in Figure 3. Reteaching merely presents the correct method for solving a task without indicating what specific error(s) were made. Figure 4 shows the process of reteaching a task.

Figure 5 illustrates the process of diagnosing and remediating students' errors with the PIXIE system, before any of the system enhancements, which are discussed later in this paper, were added. During a tutorial session, a student is given a task from the domain rulebase and produces a response. If the answer is correct, then the student moves on to the next task. Otherwise, the incorrect answer is compared to those generated in the offline phase to determine if an appropriate model has been generated. If a match is found, then a diagnosis of the student's error is hypothesized. In the remedial phase, the hypothesized model is presented to the student, who is asked if the model resembles the method he/she used to solve the task. If the hypothesis is confirmed, the student receives model-based remediation (MBR). Otherwise, the hypothesis is rejected and the task is simply retaught. If there is no match between the student's answer and any of those generated in the offline phase, then no model is presented to the student; the student is retaught the current task. In all cases, the system then moves on to the next task.

1.3 Analysis Phase

The output from the online phase, a record of a student's performance, is analyzed by the analysis phase of the system. Several different types of analyses are performed. The first involves generating overall results for each student, such as the number of tasks solved correctly, the number of tasks solved incorrectly, the number of errors diagnosed, the number of errors not diagnosed, etc. Results by class are also available for groups of

students, either for use by teachers or the system designers. The second type of analysis involves the examination of the undiagnosed errors, and possible formulation of these errors as new malrules. These malrules may then be placed "manually" in the domain knowledge base.

2. Recent Enhancements

Recent experimental use of the PIXIE system (Martinak, Sleeman, Kelly, Moore & Ward, 1987) suggested several areas in which improvement was possible. This section discusses four consequent enhancements to the PIXIE system. The first involves a reduction in the generation of redundant models during the offline phase. The second involves a more sophisticated type of remediation that can handle multiple models in the online phase. The third entails a multiple rulebase option that permits the tutoring session to switch between rulebases when necessary. The fourth is the addition of a higher-order diagnosis module that provides a more conceptual diagnosis of a student's overall performance on the PIXIE system.

2.1 Reduction in the Generation of Redundant Models

As stated above, the offline phase of the PIXIE system involves the creation of the set of all possible models for a task type, given a set of correct and incorrect rules. Previously, this phase of model generation involved the creation of many redundant models, which would later be rejected. Two models are redundant (i.e. functionally equivalent) if they involve a different sequence of rules, yet the same trace is produced in the solution of a task.

Although redundant models are eventually eliminated from the final set of models, this process of creation and elimination is an inefficient

use of computer memory because the entire set of models must be stored before it can be reduced. As the knowledge-base grows, a space limitation in the offline phase is reached, preventing the addition of new malrules that might improve diagnosis. Two changes were made to the system to avoid this memory overload. The first involves a new type of subsumption constraint, and the second a constraint on rule ordering.

SEMANTIC SUBSUMPTION CONSTRAINTS

The syntactic subsumption discussed previously involves two rules with overlapping condition sets. It is also sometimes the case that if rule r1 fires, then another rule, r2, will never fire, because r1 has produced a state from which the necessary conditions for r2 will never result. Rather than involving a shared set of conditions, as in syntactic subsumption, this type of subsumption, called semantic subsumption, involves the action of one rule being incompatible with the condition of another. Notice that an inspection of the rule conditions and actions will not necessarily indicate whether semantic subsumption is present in a model. The model must be executed on appropriate tasks before subsumption is evident.

For example, any model in which r1 (or any of its malrule variations) appears, followed by r2 (or any of its malrule variations), produces the same trace as a model without r2. Therefore, the models (r1 r2 r3), (r1 m1r2 r3), (r1 m2r2 r3), ..., will all have the same trace and so only the first need be retained. Operationally the algorithm is told that r1 semantically subsumes r2, m1r2, etc.

Semantic subsumptions were identified in the PIXIE's knowledge bases by the investigators once a set of models have been generated and executed on specific tasks. Subsequently, constraints have been implemented that prevent the creation of a known set of redundant models.

Rather than creating models, running them on a set of tasks, and then eliminating redundant models, the semantic subsumption constraints prevent the models from being generated in the first place. This consists of checking each model as it is being generated to see if it contains any rules that are known to subsume one another.

INTERACTION CONSTRAINTS

The second change aimed at reducing the number of redundant models involves the order of rules. As mentioned in the discussion of the offline phase, the order of pairs of rules is sometimes significant. In addition, groups of rules can sometimes interact, in which case models containing all orderings must be created. For example, if the rules r1, r2 and r3 interact with one another, then all permutations of those rules must be created, i.e., models (r1 r2 r3), (r1 r3 r2), (r2 r1 r3), (r2 r3 r1), (r3 r1 r2), and (r3 r2 r1). In addition, all malrule variations would be generated. Again, this may create redundancies because not all orderings may be significant. For example, models (r1 r2 r3) and (r3 r1 r2) may produce the same solution path. Previously, models were generated, evaluated and then possibly eliminated. Interaction constraints have been added that will create only the significant orderings for groups of interacting rules.

The addition of these two constraints, semantic subsumption and group interaction, has eliminated the creation of many redundant models in the offline phase. These rules do not capture all redundancies, and the elimination of models is still necessary. However, the number of redundancies has been reduced, thereby allowing the offline system to handle larger knowledge bases.

2.2 Remediation that Handles Multiple Models

When a student solves a task on the PIXIE system, a diagnosis is made if a model has been generated during the offline phase that produces the same answer as the student. Multiple models occur if more than one distinct model produces the same answer to a task. For example, Figure 6a shows two distinct methods for solving the task $3x + 4x = 21$ and reaching the same answer. Previously, multiple models could be diagnosed, but the remedial system had no mechanism for distinguishing between them. It was thought better to provide no model-based remediation, rather than possibly giving remediation for a model that the student did not use.

In order to reduce the number of multiple models, a program to generate tasks that discriminate among the greatest number of models was implemented. For instance, if given the template $ax + bx = c$, the task generator might produce the task $6x + 3x = 36$, as in Figure 6b. This task distinguishes the two solution methods used in Figure 6a. However, as the number of malrules increases, and hence the number of models, it becomes very difficult to find completely discriminatory tasks. Indeed, it is impossible to completely eliminate multiple models; in Figure 6c, it does not matter which numbers are used with these two models, the same answer will always be produced.

In order to cope with multiple models, and potentially improve diagnosis and remediation, a more sophisticated system was developed. The new system considers the student's task trace, in addition to his/her final answer, in an effort to discriminate between possible models. The new remedial procedure to handle multiple models is illustrated in Figure 7, starting at the evaluation step from Figure 5. As before, during a tutorial session, a student works a task and arrives at an answer. If the answer is correct, then the student moves on to the next task. Otherwise,

three logical possibilities can now be handled: no model, one model, or more than one model exists to describe the student's solution path. The situations in which there is no model or only one model are handled as described in the online section.

If the student has produced an answer for which there is more than one model, the problem trace for that answer is compared to each of the appropriate models. If there is only one model that could produce the student's trace, this model is presented to the student as if there had only been one model to match originally. Figure 8 illustrates the use of a student's trace to discriminate among models.

If more than one model could produce the student's problem trace, then the student is asked to rework the task showing more of his/her work. The same model-discriminating sequence is applied to the new working of the task, with one difference. If, for a second time, the student has produced an answer for which there is more than one model, and his/her problem trace does not completely discriminate among the models, then the student is not asked to rework the task again. Instead, he/she is shown a series of those models consistent with his/her own trace. Again, the student is asked whether any of the models resembles his/her own solution process. Figure 9 illustrates this procedure.

If the student's trace does not discriminate among any of the models, for example, the student types in only the answer without any intermediate steps, then several arbitrarily ordered models are presented.

An ordering function to control the order in which models are presented would improve the effectiveness of this remediation, though it has not been implemented. This ordering function could work in several different ways. One is to use information about previous errors a student has made. If a student has a tendency to make a certain type of error, and

one of the possible models contains that type of error, then that model would be assigned a higher priority for presentation than other models. Another possibility is to use the frequency of errors across a population to weight rules. If a model contains a high frequency rule or rules, then it would be presented before others of lower frequency. These two methods of ordering rules could be used alone or in conjunction with one other.

One of the drawbacks to the new remedial system is that it can create an onerous amount of text for the student to read. Also, because the student may be required to choose between task traces, it becomes even more crucial that the traces be somehow tailored to the student's style of problem solving. Often a student will answer "no" when presented with a model because there is a greater level of detail in that model than he/she explicitly uses in solving the problem, or a different style of simplification. This problem is exacerbated when the student is required to choose between models. Figure 10 lists several traces that students produced in two experiments (Martinak *et al*, 1987), along with corresponding PIXIE traces that were rejected or accepted. In Figure 10a, the only difference between the traces is the step $7x/7 = 7/6$, in which PIXIE has explicitly divided both sides by 7. In Figure 10b, the student appears to have used a "move and change the sign" approach to cancelling terms, whereas the PIXIE trace adopts the "doing the same thing to both sides approach". In addition, the third and fourth steps of the PIXIE trace explicitly include a zero. The resulting traces are quite different, and it is not surprising that the PIXIE trace was rejected. Figure 10c illustrates traces that use the same cancelling method, but the PIXIE trace includes additional zeros; our hypothesis is that these 'extra' steps led the student to reject the trace.

An experiment is being planned to explore whether the acceptability

of a trace to a student is determined by the style of simplification, the level of detail, or both. Diagnosis could then be improved by observing each student's style of simplification and choosing a method that addresses it. Despite these limitations, the current system can now handle multiple models, and is an improvement over the earlier version.

2.3 Multiple Rulebases

Algebra, like many other skills, is built upon a foundation of subskills. Several prerequisite subskills of algebra are arithmetic operations, negative numbers, precedence, and fractions. The learning of algebra may be severely hampered by a lack of understanding in any one of these subskills. Both human tutors and ITSs must cope with the complexity of interrelated skills, and varing degrees of mastery of those skills.

What would a human tutor be likely to do in a case in which a student is being tutored in algebra, and consistently makes precedence errors? For example, a student solves the task 2 * 3x + 4x = 22 as x = 22/14. A tutor may switch the focus of tutoring from algebra to arithmetic precedence until the student shows signs of understanding precedence in arithmetic, and then resume algebra tutoring. In order to focus on arithmetic precedence rules, rather than on algebraic rules, tasks of the type 3 + 4 * 5 might be given to the student.

A multiple rulebase option has been implemented in the PIXIE remedial system to emulate the ability of a human tutor to switch the focus of tutoring from one domain to another when necessary. Figure 11 illustrates the use of the rulebases for algebra and arithmetic precedence being used in a tutoring session.

The controlling data for switching between rulebases resides in a file

associated with the initial rulebase. This file contains information about errors in this rulebase that correspond to a subskill for which another rulebase exists, those levels in the second rulebase which are applicable. For example, in Figure 11 the algebra control file contained the following pieces of information that enabled the switch to the precedence rulebase:

```
model of error => (add-xterms mult)
level of error => 16
corresponding rulebase => precedence
appropriate level => 4-6
```

The online system monitors a student's performance to check if an error is made that should be tutored as a subskill, such as adding x-terms before multiplication. If so, the control file is used to switch from one rulebase to another, and back to the original rulebase after successful tutoring in the second domain. The mechanism for switching between rulebases is general and can be called recursively. That is, rulebase1 can activate rulebase2, which in turn can activate rulebase3, and so on.

One limitation to the current implementation is that the monitoring of a student's performance for certain errors is built into the online system. Implementing this function as a separate production system handler would allow for more flexibility in the use of the multiple rule bases. Educational heuristics, such as the number of items to be diagnosed before remediation should occur, could then be represented as production rules, and could vary for different errors and students.

Despite its shortcomings, the mechanism now exists for using several rulebases, and hence tutoring different subskills in the same tutoring session. This is an important step toward creating an effective computer tutor. What needs to be incorporated into the system are the heuristics

for controlling the interaction between rulebases.

2.4 Higher-order Diagnosis

Currently, the diagnosis of errors by the PIXIE system is based solely on each task as it is solved, independent of other tasks. Previous diagnosis of errors does not affect the current diagnosis. That is, PIXIE diagnosis is completely bug specific, which leads to the loss of much information that might be useful for providing better diagnosis, and thus possibly improving remediation. In general, errors do not occur in isolation, and a diagnostic system should take this into account.

Diagnosis could be made less error specific by considering the context of an error, rather than focusing solely on the error itself. The context of an error in a tutorial session includes previous diagnoses, problem type, answer type and previous tasks answered correctly.

As a first step in exploring a more context driven diagnosis, a separate module is being implemented that provides a "higher-order" diagnosis. After a student has completed an online session with PIXIE, the diagnostic program produces a summary of the student's performance. The goal of this subsystem is to create a more global conceptual diagnosis of the student's overall performance, rather than being restricted only to information about individual errors.

The initial step in the design of a more global diagnosis system was to ask two researchers on the PIXIE project to read several students' log files (a printout of the entire interaction between the student and the tutoring system), and to produce a summary of the students' misconceptions of the algebra tasks worked. One of the reseachers made the following statements:

"some difficulty understanding fractions"; (for student A)
"student may not understand the concept of 'two sides' of the equation and the balance (equality) that must be maintained". (for student B)

The summaries produced by the other researcher included:

"does not understand ax=b --> x=b/a"; (for student A)
"possibly not clear about ... x-terms occurring on both sides of the equation". (for student B)

On the basis of this analysis, a rule-based expert system for producing a diagnostic summary of a student's performance on a series of mathematical tasks has been implemented. An expert system is well suited for this task because it can weigh evidence from several sources, i.e. the several errors, and synthesize an overall result. This subsystem requires the following input:

1. domain assertions that contain knowledge about errors, categories of errors, characteristics of the different type of tasks, relationships between errors, etc.

2. a record file which is a description of a student's performance on all tasks

3. rules that will process assertions about a student's performance and produce a diagnosis.

The system is a forward-chaining rule interpreter with a front-end to process a record file to produce assertions about the student's performance. To produce a diagnosis, the system first reads a student's record file and produces a list of student-specific assertions. For instance, these assertions indicate on which task-set the student committed a certain error, or made an unknown error, etc. Secondly, the system processes these assertions about student errors, along with

domain assertions, to produce higher-level assertions about categories of errors. Thirdly, this information about different types of errors is evaluated to produce, if possible, a more global diagnosis.

The general types of misconceptions that the system addresses are:

bracket errors (distributive law)
cancelling errors (numeric and x-terms)
multiple x-terms (same side of the equation; different sides of the equation)

algebraic notation (e.g. separating the coefficient from the x-term)
fractions (proper and/or improper).

Suppose a student produces the series of traces in Figure 12a. This student has solved two types of tasks with only one x-term correctly, but incorrectly solves a task with two x-terms. From this evidence, the system concludes that the student is unable to cope with multiple x-terms. It further indicates those levels, i.e. different types of tasks, on the PIXIE system that might provide useful tutoring. In the next example, Figure 12b, the student appears to be able to solve tasks with two x-terms, if they were on the same side of the equation. Consequently, a slightly different diagnosis is offered. In the third example, Figure 12c, the student's solution strategy appeared to be influenced by the form of the answer.

The above three examples deal with task types, that is, the form of the problem that a student cannot solve. The final example, Figure 12d, addresses a more general type of error, namely, "cancelling" errors.

Notice that this system's higher-order diagnoses have a different flavour than those of the two researchers mentioned above. The researchers' comments are more general, and not related to specific tasks

and types of tasks. For instance, one researcher made the comment that a student had "some difficulty understanding fractions", whereas the system provided diagnoses of the type given in Figure 12a, and indicates the specific types of tasks with which the student has difficulty.

The current system diagnoses are intermediate between the previous system capabilities and human commentaries on student performance. The comments produced by this system tie the diagnoses more closely to specific tasks and task types, and thus may be a more useful level of comment for providing remediation.

One benefit of using this type of rule-based system for producing a diagnosis is that the rules for diagnosis are easily modifiable. The system itself need not be changed, simply the condition-action rules used in its database. The same is true of the relationship between errors and misconceptions. If, for instance, empirical results indicate that a particular error provides evidence for a student possessing a misconception, then this information can be incorporated by creating a new assertion, or modifying an existing one. Essentially, the system for diagnosis is domain independent, whereas the rules and assertions are domain dependent. This follows the general design of the PIXIE system as an ITS shell for tutoring in many domains.

The sub-system, as currently implemented, produces an overall diagnosis of a student's performance that might be useful to teachers. Most high-school teachers do not have the time to make an indepth analysis such as the enhanced PIXIE provides, and consequently may benefit from PIXIE's summary diagnosis. However, future work should aim to incorporate such context-sensitive analysis into the online system, thereby making a global diagnosis available as a basis for more sophisticated remediation. The benefit of the enhanced diagnostic system

would then directly effect remediation, and perhaps even the choice of the tasks to be presented.

If this higher-order diagnosis is incorporated into the remedial system, then the further possibility exists to develop a relationship between the new diagnostic system and the multiple rulebase option. The benefit of the multiple rulebases will be gained in judicious decisions to switch domains of tutoring. The higher-order diagnostic system may be able to provide the information necessary to switch tutoring domains on the basis of educational heuristics provided.

3. Summary

Extensive experimentation on an earlier version of the PIXIE system suggested a set of system changes that have been reported here. From an AI perspective, we undoubtedly have a more sophisticated system. Unfortunately, to date, this enhanced system has only been subjected to a limited set of field trials. The results of these have been encouraging, but only an extensive set of field trials will determine whether the enhanced system is instructionally superior.

References

Hirsh, H. B. & Sleeman, D. H. (1985). Inference of incorrect operators. In T. M. Mitchell, J. G. Carbonell & R. S. Michalski (Eds.). Machine learning: A guide to current research. (pp. 93-97). Los Gatos, CA: Kluwer.

Martinak, R., Sleeman, D. H., Kelly, A. E., Moore, J. L. & Ward, R. (1987). Studies of diagnosis and remediation with high-school algebra students. Technical Report AUCS/TR8711, Department of Computing Science, University of Aberdeen.

Sleeman, D. H. (1982). Inferring (mal) rules from pupils' protocols. Proceedings of the 1982 European A.I. Conference. (pp. 160-164). (Republished in Proceedings of the International Machine Learning Workshop, Illinois, June, 1983.)

Sleeman, D. H. (1983). Inferring student models for intelligent computer-aided instruction. In R. S. Michalski, J. G. Carbonnel, & T. M. Mitchell (Eds.). Machine learning: An artificial intelligence approach. pp483-510. 7Palo Alto, CA: Tioga.

Sleeman, D. H. (1987). PIXIE: A shell for developing intelligent tutoring systems. In R. W. Lawler & M. Yazdani (Eds.). Artificial intelligence and education, vol. 1. (pp. 239-265). Norwood, NJ: Ablex.

Sleeman, D. H. & Smith, M. J. (1981). Modelling students' problem-solving. Artificial Intelligence, 16(2), 171-187.

Figure 1. The PIXIE system.

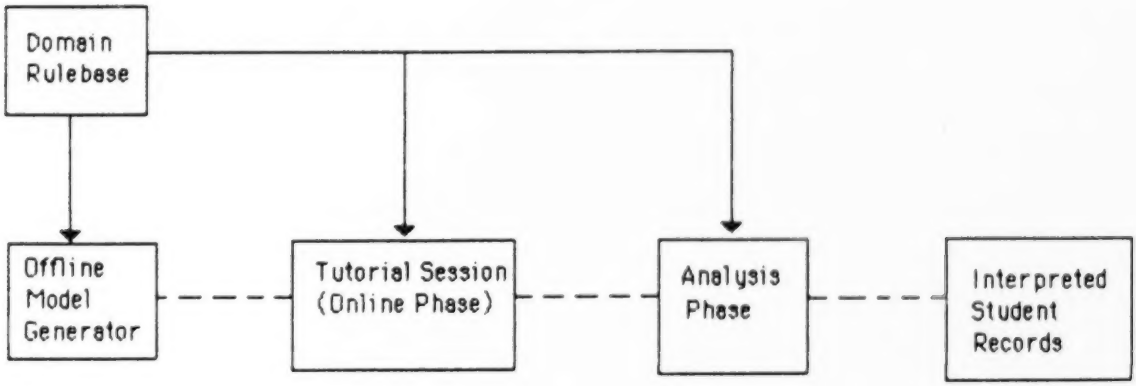

Figure 2. The use of rules and malrules in student models.

a.)

	2x + 4 = 9		2x + 4 = 9
n-to-rhs	2x = 9 - 4	mal-n-to-rhs	2x = 9 + 4
add	2x = 5	add	2x = 13
solve	x = 5/2	solve	x = 13/2

b.)

	2x = 3*4 + 5		2x = 3*4 + 5
mult	2x = 12 + 5	add	2x = 3*9
add	2x = 17	mult	2x = 27
solve	x = 17/2	solve	x = 27/2

Figure 3. Model-based remediation.

(This protocol is slightly edited for readability.)

Solve (f = Finished, q = Quit early)
7x = 2*2
7x = 14
f

Simplify further 7x = 14
x = 4
f

Thank you,
but you didn't get that one right.
We will look at that task again.

Here we are considering the task 7x = 2*2

Did you work the task something like this?

7x = 2*2
7x = 4
7x/7 = 4
x = 4
Please type YES or NO: y

It seems you changed 7x = 4 to something like 7x/7 = 4

This is WRONG
DO NOT only divide ONE side by the coefficient.
You should divide both sides of the equation by the coefficient to change
7x = 4 to 7x/7 = 4/7

Now let me show you how you SHOULD have worked the whole task.

7x = 2*2	multiply 2 numbers
7x = 4	divide both sides of the equation by the coefficient
7x/7 = 4/7	divide
x = 4/7	finished

Figure 4. Reteaching.

(This protocol is slightly edited for readability.)

Solve (f = Finished, q = Quit early)
7x = 2*2
7x = 4
f

Simplify further 7x = 4
x = 4
f

Thank you,
but you didn't get that one right.
We will look at that task again.

Now let me show you how you SHOULD have worked the whole task.

7x = 2*2	multiply 2 numbers
7x = 4	divide both sides of the equation by the coefficient
7x/7 = 4/7	divide
x = 4/7	finished

Figure 5. The Online Phase.

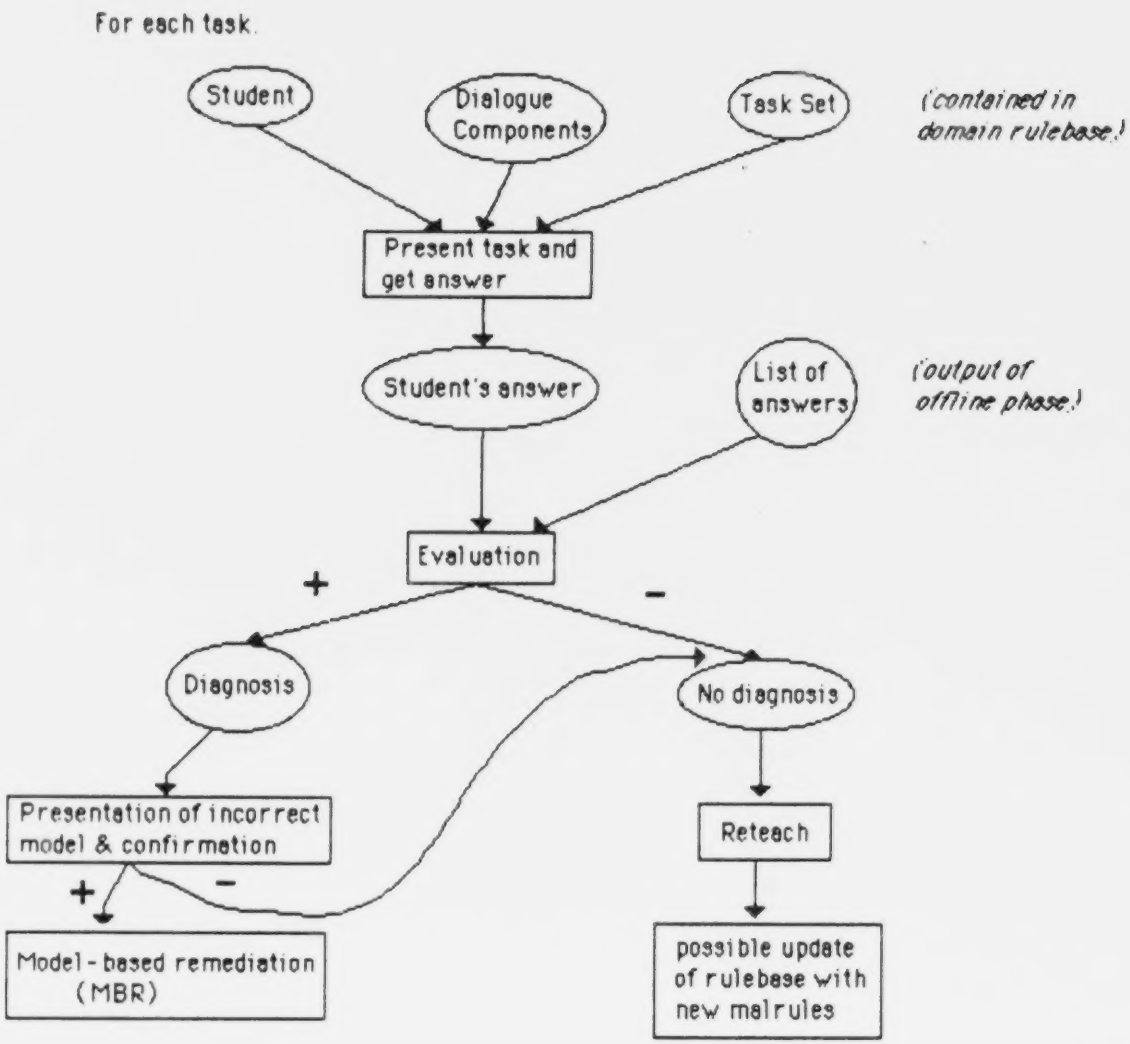

Figure 6. Examples of Multiple Models.

a.) Two methods for solving a task which happen to produce the same answer.

$3x + 4x = 21$	$3x + 4x = 21$
$x + x + 3 + 4 = 21$	$x + x + 3 + 4 = 21$
$2x + 7 = 21$	$2x + 7 = 21$
$2x = 21 - 7$	$2x = 21 + 7$
$2x = 14$	$2x = 28$
$2x/2 = 14$	$x = 28/2$
$x = 14$	$x = 14$

b.) Changing the parameters produces different answers for each method.

$6x + 3x = 36$	$6x + 3x = 36$
$x + x + 6 + 3 = 36$	$x + x + 6 + 3 = 36$
$2x + 9 = 36$	$2x + 9 = 36$
$2x = 36 - 9$	$2x = 36 + 9$
$2x = 27$	$2x = 45$
$x = 27/2$	$x = 45/2$

c.) Two methods which will always produce the same answer, regardless of the parameters.

$3x = 5$	$3x = 5$
$x + 3 = 5$	$3x/3 = 5$
$x + 3 - 3 = 5$	$x = 5$
$x = 5$	

Figure 7 Remediation to handle multiple models.

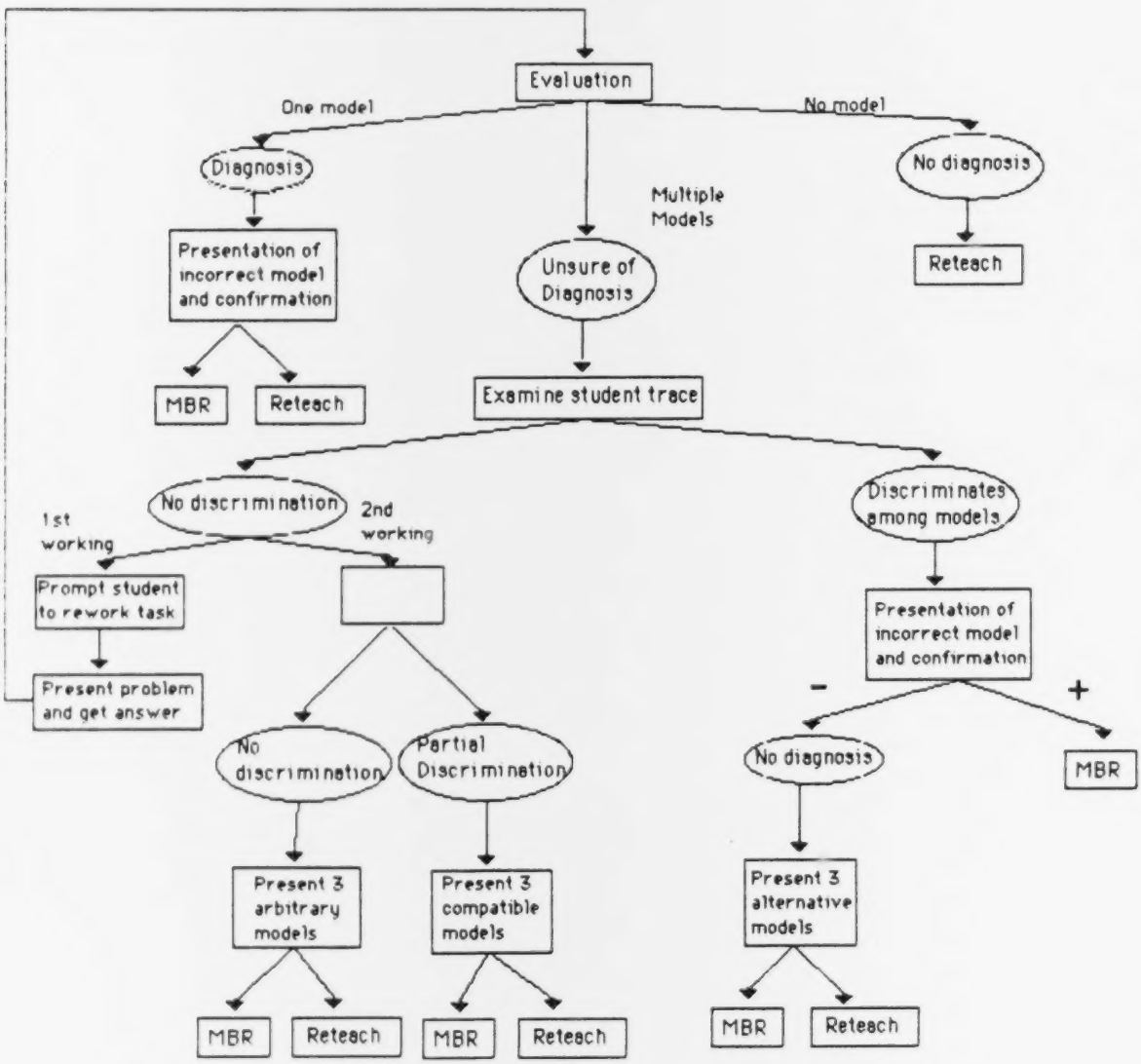

Figure 8. Use of student trace to discriminate among models.

Solve (f = Finished, q = Quit early)
3x + 4 = 19
7x = 19
x = 19
f

Thank you,
but you didn't get that one right.
We will look at that task again.

There are at least 3 ways of solving this task incorrectly and arriving at the answer 19. For instance:

3x + 4 =19	3x + 4 = 19	3x + 4 = 19
3x + 4 - 4 = 19	x + 7 = 19	7x = 19
3x = 19	x + 7 - 7 = 19	7x/7 = 19
3x/3 = 19	x = 19	x= 19
x= 19		

On the more difficult task levels PIXIE has up to 12 ways of solving a task and arriving at the same answer. In this case, because the student has typed in the one intermediate step 7x = 19, the models can be distinguished, and a diagnosis is made.

Here we are considering the task 3x + 4 = 19
Did you work the task something like this?:

3x + 4 = 19
7x = 19
7x/7 = 19
x = 19
Please type YES or NO: **y**

Model-based remediation occurs.

Figure 9. The use of a student trace to partially discriminate among models.

Solve (f = Finished, q = Quit early)
7x = 5x + 17
x = 17
f

Thank you.
Please solve that problem again and show PIXIE more of your work.

Solve (f = Finished, q = Quit early)
7x = 5x + 17
x + 7 = 5x + 17
x = 17
f

Thank you,
but you didn't get that one right.
We will now look at that task again.

PIXIE is not sure what you did incorrectly.
You will now see at most 3 ways of solving the task.
Please answer YES if one of these ways resembles
the way you solved the task.

PIXIE has 6 methods of solving this task and arriving at the answer 17; 3 of these ways involve separating the coefficient from the x-term (as the student did), and 3 of them do not. Only those models consistent with the student's trace are presented to the student in the tutoring session. The models which were NOT presented to the student are:

7x = 5x + 17	7x = 5x + 17	7x = 5x + 17
2x = 17	7x + 5x = 17	7x = 5x - 5x + 17
2x/2 = 17	12x = 17	7x = 17
x = 17	12x/12 = 17	7x/7 = 17
	x= 17	x = 17

Those models consistent with the student's trace are listed below.

Did you work the task something like this?:

7x = 5x + 17
x + 7 = 5x + 17
x + 7 - 7 = 5x + 17
x= 5x + 17

Figure 9. (cont.)

x = 5x - 5x + 17
x = 17
Please type YES or NO: **n**

Did you work the task something like this?:

7x = 5x + 17
x + 7 = 5x + 17
x + 7 - 7 = 5x + 17
x= 5x + 17
x - 5x = 5x - 5x +17
-4x = 17
-4x/4 = 17
x= 17
Please type YES or NO: **n**

Did you work the task something like this?:

7x = 5x + 17
x + 7 = 5x + 17
x + 7 - 7 = 5x + 17
x = 5x + 17
x + 5x = 17
6x = 17
6x/6 = 17
x= 17
Please type YES or NO: **y**

Student would receive model-based remediation on any model to which they answered "yes".

Figure 10. Student traces and rejected or accepted PIXIE traces.

a) Student trace:
7x = 3 * 2
7x = 6
x = 7/6

Pixie trace (rejected):
7x = 3 * 2
7x = 6
7x /7 = 7/6
x = 7/6

b) Student trace:
6x + 4 = 21
6x = 21 - 4
6x = 17
x = 6/17

Pixie trace (rejected):
6x + 4 = 21
6x + 4 - 4 = 21 - 4
6x + 0 = 21 - 4
6x + 0 = 17
6x = 17
6x/6 = 17/6
x = 6/17

c) Student trace:
6x + 4 = 21
6x + 4 - 4 = 21 + 4
6x = 25
x = 25/6

Pixie trace (rejected):
6x + 4 = 21
6x + 4 - 4 = 21 + 4
6x + 0 = 21 + 4
6x + 0 = 25
6x = 25
6x/6 = 25/6
x = 25/6

---

d) Student trace:
3x + 4 = 19
x =19/7

Pixie trace (accepted):
3x + 4 = 19
7x = 19
7x /7 = 19/7
x = 19/7

e) Student trace:
7x = 2 * 2
7x = 4
x= 4

Pixie trace (accepted):
7x = 2 * 2
7x = 4
7x/7 =4
x= 4

Figure 11. Use of multiple rulebases in a tutorial session.

Student has just worked 3 tasks of this type incorrectly and received remediation.
He/she proceeds to work the next 3 tasks using the same incorrect method.

Solve (f = Finished, q = Quit early)
2 * 3x + 4x = 12
x = 12/14
f

Let's go on to the next task

Solve (f = Finished, q = Quit early)
4 * 3x + 4x = 4
x = 4/28

Let's go on to the next task

Solve (f = Finished, q = Quit early)
2 * 4x - 2x = 12
x = 3
f

Thank you.
You got 0 of out 3 right.

We will now give you some problems in some more basic math skills.

Solve (f = Finished, q = Quit early)
2 * 3 + 4
14
f

Let's go on to the next task

Solve (f = Finished, q = Quit early)
3 * 2 + 5
21
f

Thank you.
You got 0 of out 2 right.

Figure 11. (cont.)

We will now look at the 2 tasks that you worked incorrectly.

Here we are considering the task 2 * 3 + 4
Did you work the task something like this?:

2 * 3 + 4
2 * 7
14
Please type YES or NO: y

It seems that you changed 2 * 3 + 4 to something like 2 * 7

This is WRONG
Do NOT add before multiplying
You should multiply to change 2 * 3 + 4 to 6 + 4

Now let me show you how you SHOULD have worked the whole task

2 * 3 + 4	multiply
6 + 4	add
10	finished

Similar remediation occurs for the second task.

Solve (f = Finished, q = Quit early)
2 * 1 + 7
9
f

Let's go on to the next task

Solve (f = Finished, q = Quit early)
0 * 1 + 6
6
f

Let's go on to the next task

Solve (f = Finished, q = Quit early)
3 * 5 + 6
21
f

Figure 11. (cont.)

Thank you.
You 3 out of 3 right.

Well done!!

We will now continue with more algebra tasks.

Tutoring continues with the algebra rulebase.

Figure 12. Student traces and PIXIE diagnoses.

a) Traces:

3x + 5 = 26	7 + 4x = 19	3x + 4x = 14
3x = 21	4x = 12	x + 4 = 14 - 3 -4
x = 7	x = 3	2x = 7
		x = 7/2

Diagnosis:
The student can solve tasks involving one X-term, but cannot handle tasks with two X-terms, such as ax + bx = c. Levels 7, 10, 12, 14 and 15 of the PIXIE system would be appropriate for tutoring.

b) Traces:

3x + 5 = 26	7 + 4x = 19	3x + 4x = 14	5x = 3x + 8
3x = 21	4x = 12	7x = 14	2x = 8 + 3 - 5
x = 7	x = 3	x = 2	2x = 6
			x = 3

Diagnosis:
The student can solve tasks involving one X-term or two X-terms on the same side of the equation, but cannot handle tasks with X-terms on both sides of the equation, such as ax = bx + c. Levels 10, 12 and 15 of the PIXIE system would be appropriate for tutoring.

c) Traces:

7x = 14	2x = 6	5x = 7	8x = 18
x = 2	x = 3	x = 2	x = 10

Diagnosis:
The student can solve tasks of the form ax = b, when b is divisible by a, but cannot solve tasks of this type when b is not divisible by a. Levels 3 and 4 of the PIXIE system would be appropriate for tutoring.

d) Traces:

3x = 5	5x = 4x + 8
3x + 5 - 5 = 26	9x = 8
3x = 26	x = 8/9
x = 26/3	

Diagnosis:
Student has difficulties cancelling terms in equations (constants, x-terms or both). Levels 8 and above would be appropriate for tutoring.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Enhancing PIXIE's Tutoring Capabilities		5. TYPE OF REPORT & PERIOD COVERED Final Report June 1984 to Sept 1987
		6. PERFORMING ORG. REPORT NUMBER AUCS/TR8709
7. AUTHOR(s) J L Moore D Sleeman		8. CONTRACT OR GRANT NUMBER(s) MDA 903-84-k-0297
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computing Science Department The University ABERDEEN AB9 2UB Scotland UK School of Education Stanford University STANFORD CA94305 USA		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q161102B74F
11. CONTROLLING OFFICE NAME AND ADDRESS Basic Research Office, Army Research Institute 5001 Eisenhower Avenue Alexandria, VA 22333-5600		12. REPORT DATE December 1987 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE --

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

--

18. SUPPLEMENTARY NOTES

COR: Dr J Orasanu

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Model-based remediation; global diagnosis; diagnosis; remediation; intelligent tutoring systems.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

See attached.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

This paper discusses the overall design of the PIXIE Intelligent Tutoring System, and more specifically a series of recent enhancements. The original system has been implemented to involve three separate phases: the offline, or model generation, phase, which is completed prior to any interaction with a student, involves the construction of a set of student models for a given domain. Considerable effort has been expended to ensure that these sets are complete and non-redundant. The online phase involves the tutorial interaction with a student, consisting of both diagnosis and remediation of errors. During the post-interaction analysis phase, undiagnosed errors are examined and, if consistent, added to the existing domain knowledge base. Four recent enhancements to the system are then discussed, each arising from a shortcoming that was noted in the system as a result of student trials. Two of these additions to the system involve the diagnosis of errors, and two involve the remediation of errors.

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)